SWISSBOX
REDESIGNING SYSTEMS FROM THE GROUND UP

Gustavo Alonso
Systems Group
Dept. of Computer Science
ETH Zürich, Switzerland
Systems Group = www.systems.ethz.ch
Enterprise Computing Center = www.ecc.ethz.ch
Gustavo Alonso, Donald Kossmann, Timothy Roscoe: SWissBox: An Architecture for Data Processing Appliances. CIDR 2011: 32-37
The SwissBox project

- Build an open source data appliance
  - Hardware
  - Software
- What is a DB appliance?
  - Database in a box
    - Funny database
    - Funny box
• Intelligent storage manager
• Massive caching
• RAC based architecture
• Fast network interconnect
NETEZZA (IBM) TWINFIN

- No storage manager
- Distributed disks (per node)
- FPGA processing
- No indexing
SAP HANA

- Main memory database
- Column store
- No indexing (automatic)
SwissBox themes

- System co-design
  - OS/DB co-design
  - HW/SW co-design
- Data processing on modern hardware
- New system architectures
  - Databases (Crescando, SharedDB)
  - Operating systems (Barrelfish)
  - Intelligent storage engines (Ibex)
Swissbox mantras

- Everything is a distributed system
  - Multicore = cluster
- Everything is heterogeneous:
  - Computing nodes
  - Memory
  - Links/networks
- Hardware can be tailored
- Performance must be predictable
How it all started:
The Amadeus use case
Amadeus Workload

- Passenger-Booking Database
  - ~ 600 GB of raw data (two years of bookings)
  - single table, denormalized
  - ~ 50 attributes: flight-no, name, date, ..., many flags

- Query Workload
  - up to 4000 queries / second
  - latency guarantees: 2 seconds
  - today: only pre-canned queries allowed

- Update Workload
  - avg. 600 updates per second (1 update per GB per sec)
  - peak of 12000 updates per second
  - data freshness guarantee: 2 seconds

- Problems with State-of-the-Art
  - Simple queries work only because of mat. views
    - multi-month project to implement new query / process
  - Complex queries do not work at all
Better the devil you know …

- Performance depends on workload parameters
  - changes in load (updates, columns accessed) -> huge variance
  - Unpredictable performance, impossible to tune correctly
Hardware killed the software star
Hardware dominates the game

- Hardware evolving faster than software
- Performance gains no longer for free
- Machines becoming far more complex
- Design assumptions no longer hold
  - Multicore heterogeneity
  - Hardware acceleration
  - Effect of networking
Hardware is going crazy

Each package has:
• 12 cores
• 4HT ports
• 4 memory channels

Each die has:
• 6 cores
• 4HT ports
• 2 memory channels
Multicore challenge

Example: deployment on multicores

<table>
<thead>
<tr>
<th></th>
<th>Min Cores</th>
<th>Partition Size [GB]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel Nehalem</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>AMD Barcelona</td>
<td>5</td>
<td>1.6</td>
</tr>
<tr>
<td>AMD Shanghai</td>
<td>3</td>
<td>2.6</td>
</tr>
<tr>
<td>AMD MagnyCours</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

Experiment setup
- 8GB datastore size
- SLA latency requirement 8s
- 4 different machines
Adding resources does not help

MySQL TPC-WB 20 GB DB

- 8 cores
- 24 cores
- 48 cores

Clients vs. Throughput (TPS)
Load interaction (multicore)
Load interaction (virtualized)

**System Performance - Throughput**

- **48 cores - isolated**
- **48 cores - noisy**

*What we expect to get*

*What we actually get*

---

**Experiment setup**
- AMD MagnyCours
- 4 x 2.2GHz AMD Opteron 6174 processors
- Total Datastore size 53GB
- Noise: another CPU-intensive task running on core 0

---

**Throughput [req/second]**

- **Increasing load [#requests]**
SWISSBOX: storage engine
Software part

Crescando: the Amadeus use case

- Remove load interaction
- Remove unpredictability
- Simplify design for scalability and modeling

- Treat a multicore machine as a collection of individual nodes (not as a parallel machine)
- Run only on main memory
- One thread per core
- Highly tune the code at each core
Scan on a core

DATA IN CIRCULAR BUFFER (WIDE TABLE)

BUILD QUERY INDEX FOR NEXT SCAN

QUERIES

UPDATES

READ CURSOR

WRITE CURSOR
Crescando on 1 Machine (N Cores)
Why is this interesting (industry)?

- Fully predictable performance
  - Response time determined by design regardless of load
- Only two parameters:
  - Size of the scan
  - Number of queries per scan
- Scalable to arbitrary numbers of nodes
Why is this interesting (research)?

- Storage engine with a different interface
- Can be used as intelligent (active) storage engine
- Modular component
- No multi-threading, no fancy parallelism, no synchronization, no shared data structures, etc.
- Suitable for hardware acceleration
SWISSBOX: data processing engine

SharedDB does not run queries individually (each one in one thread). Instead, it runs operators that process queries in batches thousands of queries at a time.
Shared DB can run TPC-W!
Predictability at scale

- SharedDB can run complex joins (and shorts) in predictable time with large update loads
- Linear scalability with number of processing units (cores)
Raw performance
Predictability, robustness
Why is this interesting (industry)?

- Optimize whole loads rather than individual queries/services
- Fully predictable performance
- Better use of resources and parallelism
- Eliminates complex database administration problems
Why is this interesting (research)?

- One plan to
  - Optimize
  - Deploy ➡️ NEW
  - Schedule ➡️ NEW

- DB/OS codesign:
  - The database does not know about the hardware or its state ...
Why is this interesting?

- SharedDB runs on a heterogeneous storage engine:
  - Crescando
  - Key value store

- Same idea can be generalized to different representations and/or hardware architectures - see next
SWISSBOX:

hardware acceleration

Parallel Computation of Skyline Queries
Louis Woods, Jens Teubner and Gustavo Alonso.
IEEE FCCM, March, 2013
SELECT customer_name FROM cells WHERE amount > 200;

- Only the relevant columns (customer_name and required rows)
- where amount > 200 are returned to hosts

- CPU consumed by predicate evaluation is offloaded

- Moving scan processing off the database host frees host CPU cycles and eliminates massive amounts of unproductive messaging
  - Returns the needle, not the entire haystack

1. Smart Scan Constructed And Sent To Cells
2. Rows Returned
3. Consolidated Result Set Built From All Cells
4. 2MB of data returned to server
Intelligent storage engine

- Power supply
- Electricity meter
- Laptop
- MySQL (Ibex)
- SIRC over Ethernet
- FPGA board
- SATA II
- SSD
Inserting the FPGA in the data path

MySQL Server
- SQL interface
- Management services
- Parser
- Optimizer
  - Software
  - Hardware

SSD
raw data
FPGA
filtered data
host system (MySQL)
Engine design
So far so good

- **SELECT a, COUNT(*) FROM table AS t GROUP BY a**

- **SELECT * FROM table AS t WHERE a = const**
### Points of interest

<table>
<thead>
<tr>
<th>Query/Storage Engine</th>
<th>Δ-Power</th>
<th>Energy Consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point Query / MyISAM</td>
<td>22 watts</td>
<td>864 joules</td>
</tr>
<tr>
<td>Point Query / INNODB</td>
<td>24 watts</td>
<td>7380 joules</td>
</tr>
<tr>
<td>Point Query / Ibex</td>
<td>3 watts</td>
<td>216 joules</td>
</tr>
<tr>
<td>Hybrid Join / MyISAM</td>
<td>22 watts</td>
<td>864 joules</td>
</tr>
<tr>
<td>Hybrid Join / INNODB</td>
<td>24 watts</td>
<td>7380 joules</td>
</tr>
<tr>
<td>Hybrid Join / Ibex</td>
<td>3 watts</td>
<td>216 joules</td>
</tr>
<tr>
<td>Group By / MyISAM</td>
<td>22 watts</td>
<td>864 joules</td>
</tr>
<tr>
<td>Group By / INNODB</td>
<td>24 watts</td>
<td>7380 joules</td>
</tr>
<tr>
<td>Group By / Ibex</td>
<td>3 watts</td>
<td>216 joules</td>
</tr>
</tbody>
</table>

#### CPU usage when executing GROUP BY

**INNODB**

- CPU Usage
- CPU Usage History

**Ibex**

- CPU Usage
- CPU Usage History
SWISSBOX: Challenges ahead
Execution platform very complex

- Optimization not trivial
  - Multicore
  - Load interaction
  - Virtualization
  - Lack of precise cost models
  - Not stable/standard platforms
Beyond plan optimization

- Other aspects to optimization
  - Deployment on heterogeneous architecture
  - Scheduling
    - Parallel threads
    - Across queries
  - Load interaction
Hardware dictates everything

- Many options enabled by hardware
  - Custom hardware
  - Custom configurations
  - Multiplicity computing
- Reasonable development cost
- Can beat almost any software design by tuning the hardware
CONCLUSIONS
May you live in interesting times

- Many radical changes in IT infrastructure
  - Cloud computing
  - Hardware & architecture relevant again
  - Specialization / tailoring
  - Large scale clusters
  - Geographic distribution
- Great opportunity for research